Modelling and using product architectures in mechatronic product development

Bruun, Hans Peter Lomholt; Mortensen, Niels Henrik

Publication date:
2012

Modelling and using product architectures in mechatronic product development

Hans Peter Lomholt Bruun
M.Sc.M., Ph.D.-student
DTU Mechanical Engineering
hplb@mek.dtu.dk

Niels Henrik Mortensen
Professor, Ph.D.
DTU Mechanical Engineering
nhmo@mek.dtu.dk

Abstract
The objective for the paper is to determine the role of a product architecture modelling tool to support communication and to form the basis for developing and maintaining product structures for improving development practices of complex products. This paper contains descriptions, observations, and lessons learned from a case study in which the author tested a modelling tool to represent a product's architecture during product development in a larger Danish company. The reasons leading to the use of the specific model and it's terminology is described and illustrated. The paper supports two fundamental theoretical viewpoints; *Theories of technical systems* and *theories of design processes*. In this framing, the paper addresses the engineering activity of developing products supported by product architecture representations. The paper includes the description of a visual architecture representation, experiences by using the architecture representation in a mechatronic development project, and the scope of using the architecture model as a skeleton for a data structure in a PLM system. The fundamental idea for planning and modeling holistic architectures is that an improved understanding of the whole product system, will lead to better decision making. Moreover, it is discussed how the sometimes intangible product structures within a architecture can be visually modeled based on the assumption that knowledge about a product's architecture has to be tangibly instantiated, in order for people and decision makers to successfully share it and use it. **Keywords:** *Architectural design process, product models, top-down synthesis, PLM systems.*

Introduction
In the last decades many companies have moved design and manufacturing activities to low cost countries in order to sustain productivity growth. Due to the global product development and manufacturing activities, companies experience an extended value chain which is global and often fragmented and thus creates new challenges that companies must overcome (1). The challenges of overcoming the complexity of a fragmented vale chain is supported and made possible by advances in information and communication technology. It is not new that manufacturing companies use IT-systems to support management of products in their different life phases including computer aided design, engineering, manufacturing and product management tools (2). The increasing amount of data from multiple IT-systems can however often be hard combinable because the large datasets generated by different systems have tended to remain trapped in their respective systems because of their different format and information structure. One of the strategies for handling product data from multiple IT-systems is Product Lifecycle Management (PLM). The idea behind PLM-systems is that companies can create more value and develop less waste by integrating interdisciplinary data from multiple systems. PLM can thus be seen as an integrated, information-driven strategy of managing the whole life cycle of a product starting from generating an idea, concept

description, business analyses, product design and solution architecture, technical implementation, and product testing, to the entrance to the market, service, maintenance, and product improvement (3). Product architectures are the mindset behind a product and a business which in brief describes how products and business processes are built up including the rules for designing within product projects (4). Product architectures consequently holds product information handled in PLM-systems. Moreover, architecture descriptions take place within the context of a project and/or an organisation and are performed throughout the system's life cycle. Consequently a product architecture potentially influences processes throughout the system of products' life cycle. In other words the conceptualization of a product family expressed in a product architecture model assists the understanding of the product system's essence and key properties pertaining to its behavior, composition and evolution (5). Whichever format an architecture representation has some basic concepts seems to be valid for them (6): *Displacement* – moving the world into a technology; *Abbreviation* – simplifying the complexity of the reality; and *Remote control* – handle the reality through models in order to intervene. By abbreviation, remote control and displacement it is possible to focus on a particular part of the product program or product itself. Two-dimensional inscriptions of the world have been recognized to have a strength in their format because they, among others, are mobile, scalable, can be reproduced, recombined, and be superimposed (7). The myriad of information belonging to a product family is however difficult to encapsulate and manage in a single two-dimensional representation, and detailed information belonging to the product architecture are preferable handled in IT-systems because models that have a computer-readable format, allow fast, precise, and safe data transfer, as well as reducing the effort to replicate and modify information (8). One of the challenges in managing information in a lifecycle perspective is the aspect of representing product families' multiple structures and the enormous amount of information belonging to them. How is the skeleton for the data structure model developed, ensuring a suitable alignment of technical domains in a lifecycle perspective? The approach in this study has been to use an architectural product modelling method to capture the information belonging to different product structures seen from different technical domains, and to transit the modeled structures into a PLM system in which the structures have been enriched with detailed information that is not possible to represent in a two-dimensional product architecture model.

Approach/research strategy

The theoretical basis or the fundamental viewpoint from which this article sets off has its cornerstone within engineering design science and it has a systems and design perspective on the topic of architectures and product platforms. In this framing, the article addresses the engineering challenge of modelling product architectures to support development of mechatronic product families and how models and their information can be handled in PLM systems. Mechatronics is in this paper regarded as a multidisciplinary field of engineering, with an approach aiming at the optimal integration of mechanics, electronics, control theory, and computer science within product design and manufacturing. Problematisation in this context considers the concrete elements of modelling and capturing essential information and thereby supporting development of product families and management of the process in manufacturing companies. This paper describes a part of an action research study conducted in a large-sized high-tech company in which the developed architecture model has been implemented. This article will focus on describing the modelling formalism and the process for creating and maintaining it, as well discussing if the model helped to an improved understanding of the whole product system and if it could be measured that it lead to better decision making.

Modeling product architectures to support mechatronic product development

The approach of developing complex products or entire product families can be supported by using product architecture models in which high level descriptions improve multidisciplinary communication and cooperation (9-11). Simple products may have little use for architecture descriptions but when developing complex products, it is a prerequisite to have a superimposed view on technical domains in order to foresee performance in the products' meetings with its life phases. The product architecture model presented in the following is named *The Interface Diagram* (IFD). The IFD has its basis in the *Generic organ diagram* presented in the work of Ulf Harlou (9). The model represents a product architecture comprehended as a structural characteristic of a system, mostly combining an aspect of mapping between technical domains, and mainly a mapping between function and physical structure. The model puts emphasise on managing technical interfaces between entities in the model, hence the chosen name of the modelling tool. Moreover the model puts emphasize on handling a family of products seen in different structural viewpoints. The main viewpoint is a system perspective i.e. the perspective that deals with the product's main functions or its related lifecycle. The second viewpoint is a modular viewpoint in which systems are split and components are physically joined and encapsulated into modules with simple interfaces.

Modelling formalism

The following section describes the modelling formalism, the process for creating and maintaining the modeled architecture, and elaborates on the IFD as a conceptual system model in order to allow intervention. Because of secrecy issues for the company involved, the formalism is illustrated and described by using an example of an IFD conducted for a product family of *Bobcats.* The Bobcat is a mobile power loader that is a small, self-propelled utility vehicles used for a large variety of purposes such as heavy duty agricultural earth moving, construction site utility and integrity support of public and private infrastructure. The IFD is modeled by means of blocks and lines in the software program Microsoft Visio. The program is suited for object oriented modelling and structuring different perspectives of architectures in layers. The interface diagram is normally printed on large blue prints in order to get the overview of the architecture it represents. Figure 1 is a symbolic representation of an IFD.

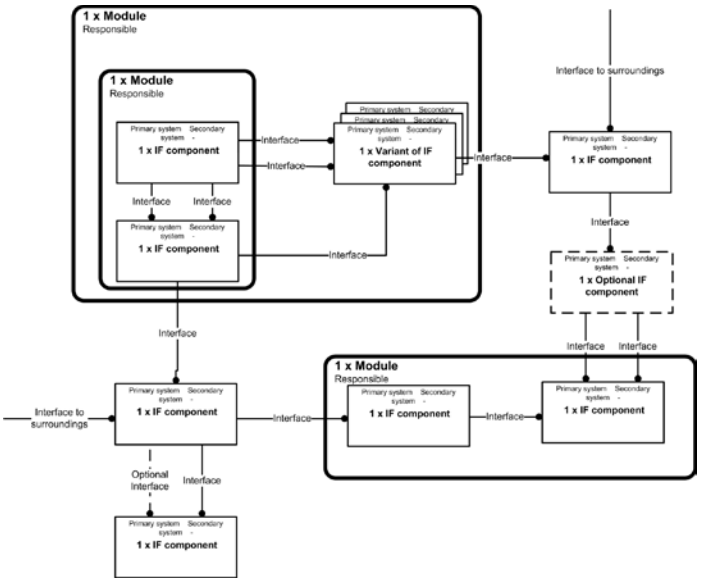

FIGURE 1 SYMBOLIC REPRESENTATION OF A GENERIC INTERFACE DIAGRAM

The diagram follows the approach of modelling architectures by use of block diagrams. The main elements of the diagram formalism are functional objects denoted Interface components (*IF-component*). The purpose of the IF-components is to decompose the systems and modules into smaller functional building blocks. IF-components can have different characteristics and are thus modeled in different ways, see Figure 2. An existing IF-component represents a generic assembly or part in the architecture and is symbolized by a white block. An optional assembly or part is symbolized by a grey block. A future assembly or part which has to be taken into consideration in the architecture but has not been developed yet has a dotted line around a white block. Finally variety of assemblies or parts is modeled by placing blocks on top of each other with a little offset. The representation shows that the specific IF-component has at least two variants. Each IF-component belongs to a product system. To avoid confusion, systems and their interaction must be clearly defined. This is done by choosing the relevant interaction as a basis for determining the system boundary. There is no fixed list of systems to be included in the development. Systems can be modeled and thereby control important properties of the final product. Each block has a designation for its system relationship; *primary* or *secondary*. A primary system means that the system designs the underlying parts and holds the responsibility for the functionality, while a secondary system has requirements to the design. An example could be that a cooling unit is designed by the team responsible for cooling and conditioning, while the system responsible for the engine has requirements to the cooling performance they need for their engine design.

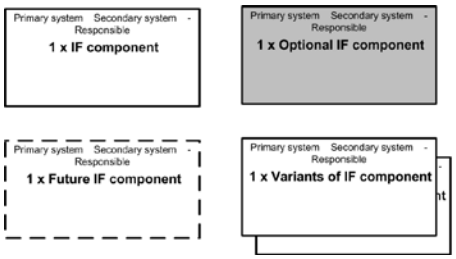

FIGURE 2 FOUR REPRESENTATIONS OF A IF COMPONENT

Modules are modeled by arranging IF-components inside boxes with a thick black boundary and rounded corners; see Figure 1. All modules are assemblies of physically joined components forming one bill of material (with possible multiple levels). Modules can contain smaller modules, but they do not overlap as it is clearly defined to which modules any element in a product belongs. Modules are in this context suitable for splitting responsibility in the product development process. The structure of the interface diagram appears as the relations are added to the diagram. The interfaces between IF-components are drawn with lines which represents a relation. An interface among two IF-components represents a physical relation, e.g. physical connection, energy transportation, information flow or flow of material. The purpose of working with interfaces is to ensure responsibility for the components interaction and to ensure that components are interchangeable, when relevant. In a product modelling context, the interface has to belong to a common structure, or some sort of generic placeholder, in order for the interfaces to be inherited to the involved elements. An interface is thus a relation between two IF-components and in every set of IF components it is defined which is the master and the slave. The dot at the end of the line indicates the responsible of the interface. This enables a responsible for an IF-component to monitor whether he owns the right to change or modify the related interface. In a modularisation context all interfaces between different modules and the outer environment should be carefully specified in order to complete the module. As described by (9) product families have a class of interfaces named generic interfaces. The generic interface are those that enable

modules to be reused and/or substituted in order to work together and these have to be stable over generations in order to enable reuse across product families and generations of product families. Optional interfaces can be modeled on the diagram to show affected relations between entities or systems if the interface is established. Figure 3 shows the IFD for the Bobcat product family with explanations of the overall layout and content. In order for the reader to quickly understand the structure of the product it is suitable to model the diagram so the layout is established as a cross section of the actual product. In that way the physical layout of the product is possible to recognize in the diagram.

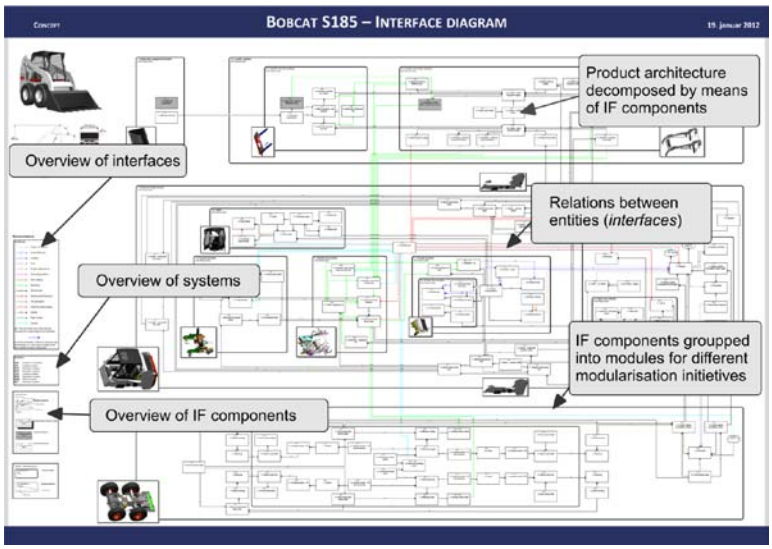

FIGURE 3 REPRESENATION OF A BOBCAT AND ITS INTERFACE DIAGRAM

The diagram can be read following the interfaces. For products that process or transforms objects this gives a logical reading direction, for other products it is up to the reader to find a suitable flow in the model.

Bridging from system to modular engineering

There exists many ways or views to read and structure a product or a family of products. According to (12):"*The structure of a product is the way in which its elements are interrelated in a system, based on the actual viewpoint"*. Consequently, a product has multiple structures depending on the viewpoint. A major strength of the IFD is its ability to handle the development of a product or product family in different lifecycle perspectives forming different structures. For highly complex products with myriads of systems, groups of specialized engineers develop entire systems. Systems are however seldom the most appropriate way of manufacturing the product and combining system after system. Modularisation in respect to aspects of assembly, transportation, sourcing, serviceability, changeability and other drivers for modularisation are thus desirable to handle in the architecture model. Modularisation creates clear structures by breaking down huge systems to manageable units, encapsulating details in larger units. Encapsulation in larger units allows descriptions to move up one level of abstraction. This reduces perceived complexity because the level of abstraction i.e. the level of encapsulation of functional units in larger chunks, creates an simpler interface to other units. An example from the Bobcat is used to illustrate this. Figure 4 shows the architecture of two of the systems in the Bobcat; the hydraulic system and the drive train system. The two systems are physically allocated to different sections of the Bobcat connected by interfaces as hydraulic hoses, electrical wires, transmissions belts etc.

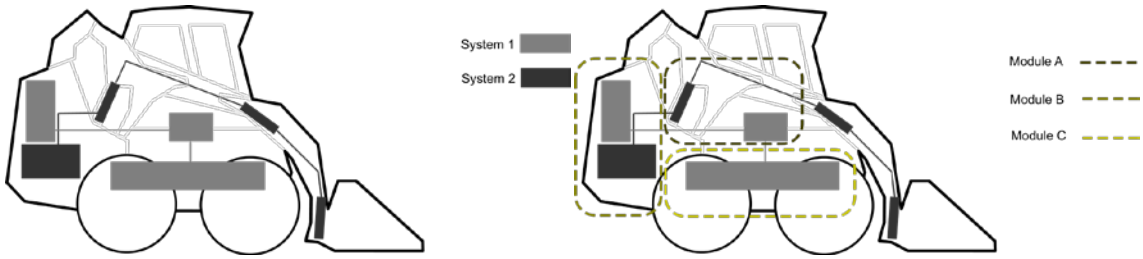

FIGURE 4 ON THE LEFT; SYSTEM STRUCTURE OF A BOBCAT SHOWING ENTITIES BELONING TO SYSTEMS AND THE RELATIONS BETWEEN THEM. ON THE RIGHT; MODULE STRUCTURE IN A BOBCAT SHOWING ENTITIES BELONING TO MODULES AND RELATIONS BETWEEN SYSTEMS AND MODULES

Modules can consist of elements belonging to different systems i.e. developed by different system teams. Figure 4 shows the architecture in which the two systems are split out in three modules: Baseframe module, engine module, and hydraulic module. It is therefore important both to integrate systems in modules, but also to handle interfaces created along the module boundaries splitting systems. When monitoring the IFD in Visio it is possible to turn on layers containing specific systems and the interfaces belonging to it, and also monitoring interfaces to other systems. Moreover it is possible to see entities belonging to modules with the same respect to interfaces. Figure 5 illustrates this way of using the IFD to hold information from multiple product structures. The functionality of using the structures from the IFD in a PLM-system makes it furthermore possible to link data from e.g. CAD to the structures. Moreover CAD structures can be monitored in a system or module context rearranging the components according to the chosen viewpoint.

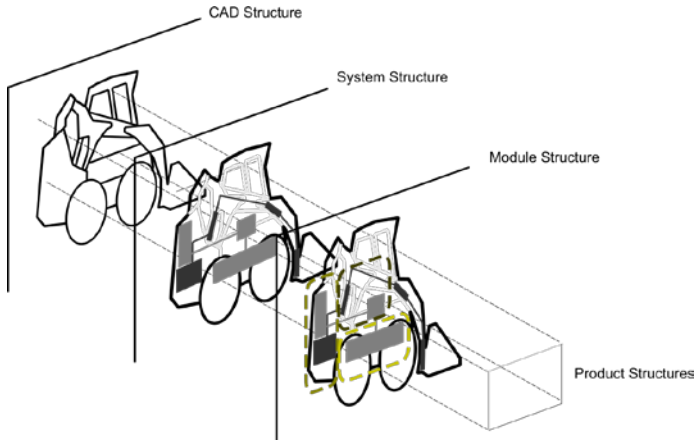

FIGURE 5 THE IFD HANDLES MULTIPLE PRODUCT STRUCTURES

Experiences from implementation and use

The interface diagram is a dynamic tool which is updated and refined during the project life cycle. The technique for modelling the architecture is based on interviewing the persons with the insight to model systems in their totality. It is seldom that a single person in a company has the needed insight to draw the diagram. Therefore several domain experts have to be involved in giving input to the diagram. Experience from the case company showed that it took several months to create a meaningful diagram. The consumed time for establishing an IFD is however depended on the complexity of the products i.e. number of different systems, and number of necessary IF-components to decompose them. The process of establishing and maintaining an IFD for a product family is thus an iterative process. The diagram should provide all systems with the holistic overview of the products' present status and not least of all, to enhance the parallelism among system development activities. The interface diagram

must thus be reviewed in an established periodic change management process. When using the interface diagram as a tool for modularisation, it is experienced that early modularisation gives more freedom to explore alternative concepts. The interface diagram enables a proactive approach for developing and evaluating alternative modular architectural concepts for the family of products it represents, because boundaries around modules can be evaluated when the freedom to design is high and the product structure is still to be fixed. The IFD formed the basis for documenting the architecture and for managing the responsibility of systems and modules and their interfaces. The IFD has in that way proven its worth for creating parallelism of development activities as well as distributing responsibilities for them. Moreover the model supported detailed development of systems and especially the integration of systems. The functionality of visualizing the product family's system and module structures in one superimposed view was recognized by the system and module managers as a way of running development activities in a more parallel than what had been possible in previous projects. One of the most important functionalities of the IFD is however that the modeled product structures can be loaded directly into the companies PLM system and form the skeleton for the information data structures. This created a more logical structuring of the information belonging to the products, which in the case company lead to more easily navigation in the IT-system and consequently leading to optimisation of the information management processes.

Conclusion

Two major ways of modelling product architectures exists: C*omputer modelling* efforts with an intention to build software computer models, such as those supported by PLM systems or configuration systems and *phenomenon models* describing the concept of product architectures from a relatively theoretical standpoint (13). The product architecture modelling method described in this article belongs to the group of phenomenon models, with a format that enables its information to transfer to a computer model. The method for modelling product architectures in block diagrams mapping the functional entities of a product and allocation of structure to functions is not new. There is nevertheless still a need in industry for modelling architectures that represents different types of information and data elements in a way that address relations to the product family's life phase systems. The experience from implementation of the IFD is that an optimum of a visual representation provides the necessary details and, yet, still maintains the overview in order to support and improve decision-making. The IFD supports the balance between creating overview and operational handling of architectures with the higher level of information they impose. Choosing the right level of decomposing systems and interfaces in the IFD is consequently of great importance for the productivity of the tool. There is no clear answer to what the appropriate decomposition level in the IFD is, given that it is contextual to the product family it represents depended on number of variants and technical complexity. The model deals with transformation of the product over time and it offers a fundamental approach for supporting proactive modular architecture development by modelling the relational aspects of modules. The product structures established in the architecture could be loaded directly into a PLM-system which enabled to manage product information on a system, module and interface level during development. The most important reason for handling the architecture model in a PLM-system is that it enables the possibility of managing the large amount of design information belonging to a mechatronic product or entire product family. The fundamental idea for using the architecture model was that an improved understanding of the whole product system, would lead to better decision making. The experience by using the model was that it acts as a vehicle for communication between stakeholders, which enabled to foresee

implications of designs and especially design changes. An area of focus in future work is the possibility of using a process structure view point in the IFD in order for enhancing the fit between design and production phases. Moreover emphasise in the ongoing research is on developing the IFD-formalism to support the evaluation of the optimal module boundaries.

References

(1) Manyika J, et. al. Big data: The next frontier for innovation, competition, and productivity. 2011.

(2) Stark J. Global product: Strategy, product lifecycle management and the billion customer question. : Springer Verlag; 2007.

(3) Sääksvuori A, Immonen A. Product lifecycle management. : Springer Verlag; 2008.

(4) Meyer MH, Lehnerd AP. The ¤power of product platforms building value and cost leadership. New York: The Free Press; 1997.

(5) IEEE.
ISO/IEC 42010 *Systems and software engineering — Architecture description*. 2011.

(6) Cooper R. Formal organization as representation: remote control, displacement and abbreviation. Rethinking Organization: new directions in organization theory and analysis.London: Sage 1992:254–72.

(7) Latour B. Visualization and cognition. Knowledge and society 1986;6:1-40.

(8) Bergsjö D, Catic A, Malmqvist J. Towards Integrated Modelling of Product Lifecycle Management Information and Processes. Proceedings of NordDesign 2008 2008:253-264.

(9) Harlou U. Developing product families based on architectures contribution to a theory of product families. Lyngby: Department of Mechanical Engineering, Technical University of Denmark; 2006.

(10) Martin MVV. Design for variety: developing standardized and modularized product platform architectures. Research in engineering design 2002;13(4):213-235.

(11) Ulrich K. The role of product architecture in the manufacturing firm. Research Policy 1995;24(3):419-440.

(12) Andreasen MM, Hansen CT, Mortensen NH. The Structuring of Products and Product Programmes. 1996.

(13) Pedersen R. Product Platform Modeling. Department of Mechanical Engineering, Technical University of Denmark, 2009.